IN THE SPECIFICATION

Please amend paragraph 0017 of the published application, as follows:

A state of a program at any time relates to the ~~sae~~ state of variables, memory addresses, and particular registers at that moment. Debugging involves observing the state of the child program at points convenient to the programmer. Such observations are usually made by stopping program execution to observe the state of the child program. This technique is known as breakpoint debugging. Breakpoint debugging involves the use of a breakpoint facility provided by the debugger to stop the execution of the child program. Breakpoints are usually associated with lines of code, and program execution temporarily stops when the relevant line of code is reached.

Please amend paragraph 0033 of the published application, as follows:

Changes to an existing debugger are as follows. A user-interface for the debugger provides a user with an option to specify how the flow is to be altered. The debugger is modified to save the instruction at the point from where the jump is to be made. The debugger inserts exception-throwing instructions at locations from where the jump is to be made, and distinguishes these exceptions as jump point exceptions rather than that of normal breakpoints. Finally, the debugger alters the instruction point to the destination address specified by the user.

Please amend paragraphs 0071- 0074 of the published application, as follows:

Step **440**   The debugger determines if the break took place because of a jump
                        point. If so, the original instruction that was saved earlier is restored in
                        the memory location. The program is single stepped.

~~Step 450~~   ~~The program is single stepped.~~

Step **450** ~~460~~   The break instruction is again reinserted at the `memset` statement so
                        that the next time the debugger can be informed that the jump point is
                        reached.

Step **460** ~~470~~   The instruction pointer register is then changed to the address of the
                        instruction which is the first statement inside the `while` loop.